PCI-8602

多功能 DA 输出卡

使用说明书

2004/02

一 概述

PCI-8602 是一款 PCI 总线多功能 DA 输出卡 ,为用户提供了 4 路模拟信号输出通道 ,2 路反相的 PWM 数字信号输出通道 , 10Bit TTL 数字量输入通道,10Bit TTL 数字量输出通道,24bit(OPTO-22 兼容) 数字量输入/输出通道。

二、性能和技术指标

2.1 性能

- 4 路模拟信号输出通道
- 模拟信号输出分辨率 16Bit
- 模拟信号输出范围[-10 ， +10]V
- PWM 输出可调范围 14bit
- PWM 输出可调精度 0.1uS
- 2 路 PWM 反相输出
- 10Bit DI TTL 兼容数字量输入
- 10Bit DO TTL 兼容数字量输出
- 24bit(82C55A)数字量输入/输出通道
- 一路数字量输入上升沿中断

2.2 模拟信号输出技术指标

- 模拟信号输出通道: 4
- 模拟信号输出分辨率 16 位
- 模拟信号输出范围: ±10 V
- 数模转换器件 DAC7744
- 建立时间: 10uS
- 精度: 优于±0.003%满量程
- 线性度: ± 1/2 LSB
- 复位状态: 0 输出
- 输出负载能力: ±5mA

2.3　ＰＷＭ 输出

• PWM 输出通道:	2 路反相
• 脉冲周期范围：	0.1uS – 1.6384mS
• 高电平输出范围:	0.0uS – 1.6384mS
• 分辨率	0.1uS
• 建立时间:	1uS
• 输出电平：	TTL 兼容
• 低电平输出：	灌电流 8mA 时，最大输出 0.5 V
• 高电平输出	源电流 0.05mA 时，最小输出 2.4V
• 复位输出	PWMA 输出 0 ，PWMB 输出 1

2.4　数字量输出

• 通道:	10 路
• 输出电平：	TTL 兼容
• 低电平输出：	灌电流 8mA 时，最大输出 0.5 V
• 高电平输出	源电流 0.05mA 时，最小输出 2.4V
• 复位输出	输出 0

2.5　数字量输入

• 通道:	10 路
• 输入电平：	TTL 兼容
• 低电平输入：	最大 0.8V
• 高电平输入	最小 2.0V

2.6　数字量输入/输出

• 结构	82C55A
• 兼容	OPTO－22
• 通道:	24 路
• 输入电平：	TTL 兼容
• 低电平输入：	最大 0.8V
• 高电平输入	最小 2.0V

2.7　数字量中断

• 通道:	1 路
• 引脚:	DI9(共用)
• 输入电平：	TTL 兼容
• 低电平输入：	最大 0.8V
• 高电平输入	最小 2.0V
• 方式:	上升沿

三、引脚描述

3.1 CN1

CN1

引脚	信号	引脚	信号
37	AO0	19	AGND
36	AO1	18	AGND
35	AO2	17	AGND
34	AO3	16	AGND
33		15	AGND
32	DGND	14	
31	DO0	13	DI0
30	DO1	12	DI1
29	DO2	11	DI2
28	DO3	10	DI3
27	DO4	9	DI4
26	DO5	8	DI5
25	DO6	7	DI6
24	DO7	6	DI7
23	DGND	5	VCC
22	DO8	4	DI8
21	DO9	3	DI9
20	PWMB	2	DGND
		1	PWMA

模拟量输出引脚 ： AO(0..2)
数字量输出引脚 ： DO(0..11)
数字量输入引脚 ： DI(0..11)
PWM 输出引脚 ： PWMA ， PWMB
数字地 ： DGND
模拟地 ： AGND
中断输入 ： DI9

3．2 辅助数字量引脚 CN2

PC7	1	2	DGND
PC6	3	4	DGND
PC5	5	6	DGND
PC4	7	8	DGND
PC3	9	10	DGND
PC2	11	12	DGND
PC1	13	14	DGND
PC0	15	16	DGND
PB7	17	18	DGND
PB6	19	20	DGND
PB5	21	22	DGND
PB4	23	24	DGND
PB3	25	26	DGND
PB2	27	28	DGND
PB1	29	30	DGND
PB0	31	32	DGND
PA7	33	34	DGND
PA6	35	36	DGND
PA5	37	38	DGND
PA4	39	40	DGND
PA3	41	42	DGND
PA2	43	44	DGND
PA1	45	46	DGND
PA0	47	48	DGND
VCC	49	50	DGND

82C55 端口 A	：	PA
82C55 端口 B	：	PB
82C55 端口 C	：	PC
数字地	：	DGND

四．寄存器描述；

本章描述了 PCI8602 所用的寄存器，如果你对 PCI 设备编程比较熟悉，或者你是非 windows 平台的客户，你可以使用这些寄存器直接对 PCI8602 编程。

PCI8602 使用的 PCI 桥控制器是 PLX 公司的 PCI9052，你可以从 PLX 公司的网站得到它的资料（www.plxtech.com）;PCI9052 根据 PCI2.1 协议，配合操作系统在初始化时完成 PnP 的的功能，为 PCI8602 卡分配必须的 IO 地址。

PCI8602 占用连续的 64 个 IO 地址，所有的 IO 操作都是 16 位的

偏移	功能	描述
0X0	A00	模拟量输出通道 0
0X2	A00	模拟量输出通道 1
0X4	A00	模拟量输出通道 2
0X6	A03	模拟量输出通道 2
0x10	DI/O	数字量输入/输出
0X12	HTime	PWM 高电平时间
0x14	ATime	PWM 周期时间
0x20	PortA	82C55 的 A 端口
0x22	PortB	82C55 的 B 端口
0x24	PortC	82C55 的 C 端口
0X26	PortCFG	82C55 的控制端口

4.1 模拟量输出寄存器；

Offset 0 ,2 ,4 ,6

分别对应模拟量输出 A00 ， A01 ， A02 ,A03 通道

写入的值范围是（ 0 ，0xffff）对应的输出（－10V ，＋10V）；

Vo＝（Value-0x8000）*20 / 0x10000

数值	输出电压	数值	输出电压
0	-10.0000V	0x9000	1.2500V
0x4000	-5.0000V	0xa000	2.5000V
0x7000	-1.2500V	0xc000	5.0000V
0x8000	0V	0xffff	10.0000V

4．2 数字量输出/输入 寄存器。

Offset 0x10

读操作时对应模拟量输入通道 DI[0..9]

写操作时对应模拟量输出通道 DO[0..9]

4.3 PWM 高电平时间寄存器 HTime

Offset 0x12;

写入的值范围是（ 0 ，0x3fff）.

PWM 高电平=Value*0.1 uS

对应 PWMA 一个周期中输出高电平的时间

4.4 PWM 周期时间寄存器 ATime

Offset 0x14;

写入的值范围是（ 0 ，0x3fff）.

PWM 周期=（Value＋1）*0.1 uS

通过对 PWM 周期寄存器 ATime 和 PWM 高电平寄存器 Htime 的操作,可以产生频率，占空比都可调 PWM 波形;

例如:要产生 5kHZ ，占空比为 50% PWM,

ATime=（200uS/0.1uS）－1 =1999;

Htime=100uS/0.1us =1000;

4.5 82C55A 端口寄存器 PortA,PortB,PortC.

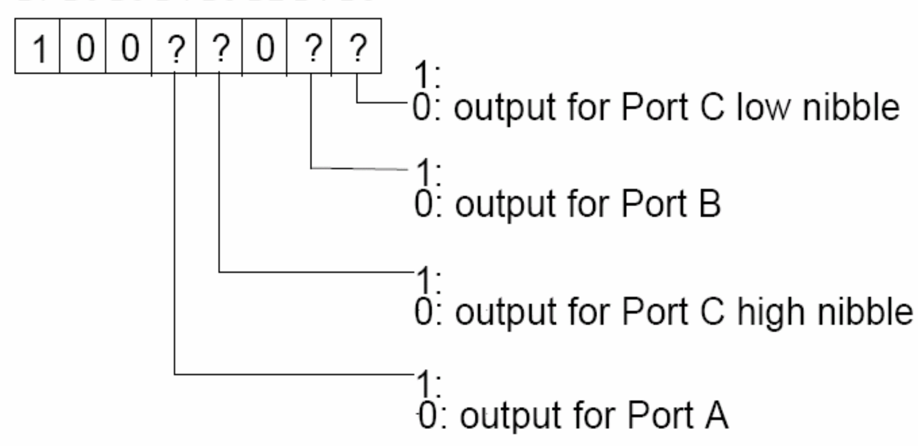
Offset 0x20 ,0x22,0x24

PortA,PortB,PortC 每个端口都是 8 位，可以编程为输入或输出；PortC 也可以编程为两个 4 位的端口。

4.6 82C55A 控制寄存器 PortCFG。

Offset 0x26;

此寄存器决定 PortA,PortB,PortC，PortC_U,Port_L 的输出/输入



如果只作简单的输入/输出各种可能的组合如下图描述

Config. Value	D 4	D 3	D 1	D 0	PORT A	PORT C UPPER	PORT B	PORT C LOWER
80H	0	0	0	0	O/P	O/P	O/P	O/P
81H	0	0	0	1	O/P	O/P	O/P	I/P
82H	0	0	1	0	O/P	O/P	I/P	O/P
83H	0	0	1	1	O/P	O/P	I/P	I/P
88H	0	1	0	0	O/P	I/P	O/P	O/P
89H	0	1	0	1	O/P	I/P	O/P	I/P
8AH	0	1	1	0	O/P	I/P	I/P	O/P
8BH	0	1	1	1	O/P	I/P	I/P	I/P
90H	1	0	0	0	I/P	O/P	O/P	O/P
91H	1	0	0	1	I/P	O/P	O/P	I/P
92H	1	0	1	0	I/P	O/P	I/P	O/P
93H	1	0	1	1	I/P	O/P	I/P	I/P
98H	1	1	0	0	I/P	I/P	O/P	O/P
99H	1	1	0	1	I/P	I/P	O/P	I/P
9AH	1	1	1	0	I/P	I/P	I/P	O/P
9BH	1	1	1	1	I/P	I/P	I/P	I/P

关于 82C55A 更详细的使用描述可以在 intrsil 公司的网站上找到。

五．硬件安装

1．关掉计算机及所有外设电源。
2．打开机盖，选择一个空槽，拧开档条的固定螺丝。
3．小心插入 PCI8602 卡，上紧档条上的螺丝。
4．盖上机盖，连接好外围设备，打开电源。

六．软件安装

6．1 软件列表

随机的软件包括驱动程序，应用程序开发库，例子程序。

驱动程序	.\Driver\Pci8602.sys .\Driver\P8602.inf
开发支持	.\inc\Pci8602.h .\Lib\PlxApi.dll .\lib\PlxApi.lib
例子程序	.\Sample*.*

6.2 驱动安装

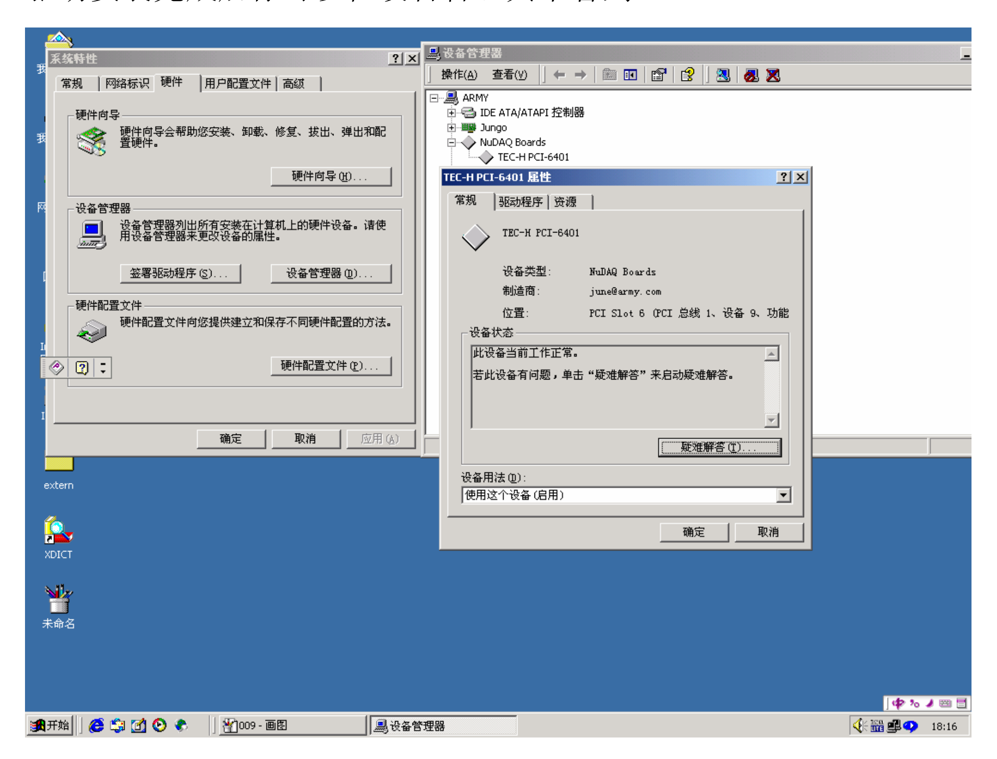
将以上文件拷贝入用户的目录，然后按以下步骤安装驱动。

依据上一章，将卡插入电脑后，打开电源。

Windows 会显示找到新的硬件，按照提示一步步超作。

当需要选择驱动时选中 P8602.inf。

驱动安装完成后你可以在设备管理其中看到 PCI8602。



七．编程指南

在 Windows 环境下，用户层应用程序不能对端口直接进行读写，必须借助 Windows 驱动程序。我们的开发包为用户提供了一组常用函数，以便于客户对板卡的操作，实现 PCI8602 的功能。

7．1 函数简介

函数名	功能简介
Regist_Pci8602	分配资源，准备调用驱动
Close_Pci8602	释放资源，结束调用驱动
AO_OutputChannel	模拟量输出函数
DO_Output	数字量输出函数
DI_Input	数字量输入函数
PWM_Output	PWM 输出设置函数
Port_RD	82C55A 的端口读
Port_WR	82C55A 的端口写
InterruptEnable	中断使能
InterruptWait	中断等待

7.1.1 初始化函数

int EXPORT Regist_Pci8602(void);

功能：　初始化 PCI8602，分配资源，准备调用驱动。

说明：　在对板卡操作之前必须首先调用此函数。

返回值：　＝1 调用成功

　　　　　＝0 调用失败

7.1.2 结束函数

int EXPORT Close_Pci8602(void);

功能：　释放资源，结束调用驱动

说明：　如果以后不再对板卡操作，调用此函数。

返回值：　＝1 调用成功

　　　　　＝0 调用失败

7.1.3 模拟量输出函数

```
int EXPORT AO_OutputChannel
           (unsigned short Channel,unsigned short Value);
```

功能：　在指定的通道上输出需要的电压值

参数：　Channel　通道编号 0，1，2，3

　　　　Value　对应电压值范围（0 ，0xffff）

返回值：　＝1 调用成功

　　　　　＝0 调用失败

7.1.4 数字量输出函数

```
int EXPORT DO_Output( unsigned short Value)
```

功能：　　在数字量输出通道输出需要值
参数：　　Value 要输出的 10bit 值，对应 DO 的[DO0..DO9]
返回值：　＝1 调用成功
　　　　　＝0 调用失败

7.1.5 数字量输入函数

```
int EXPORT DI_Input( unsigned short *Value)
```

功能：　　从数字量输入端口读入值。
参数：　　*Value 读入的值放入的地址
返回值：　＝1 调用成功
　　　　　＝0 调用失败

7.1.6 **PWM** 输出设置函数

```
int EXPORT PWM_Output
    (unsigned short TValue , unsigned short HValue )
```

功能：　　设置 PWM 输出的周期，占空比
参数：　　TValue　PWM 周期时间　＝（TValue＋1）×0.1uS
　　　　　HValue 周期中高电平时间＝HValue×0.1uS
返回值：　＝1 调用成功
　　　　　＝0 调用失败

7.1.7 82C55A 端口写函数

```
int EXPORT Port_WR (unsigned short cn_port, unsigned short  Value)
```

功能：　　82C55A 端口写
参数：　　cn_port 82C55A 的端口号.
　　　　　Value　待写出的值，对应 DO 的[DO0..DO7]
返回值：　＝1 调用成功
　　　　　＝0 调用失败

7.1.8 82C55A 端口读函数

```
int  EXPORT Port_RD (unsigned short cn_port, unsigned short  *Value)
```

功能：　　82C55A 指定端口读入值。
参数：　　cn_port 82C55A 的端口号
　　　　　*Value 读入的值放入的地址
返回值：　＝1 调用成功
　　　　　＝0 调用失败

7.1.9 中断使能函数

int EXPORT InterruptEnable(void);

功能：　　使能数字量（DI9）输入中断。

参数：　　NULL

返回值：　=1 调用成功

　　　　　=0 调用失败

7.1.10 中断使能函数

int EXPORT InterruptWait(void);

功能：　　等待中断的产生。

参数：　　NULL

返回值：　=1 中断事件产生

　　　　　=0 调用失败

注：此函数直到中断事件产生时才返回，否则一直等待。参考例子程序 **DemoInt**。

7.2 范例程序功能操作;

```
#include "stdafx.h"
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include "PCI8602.h"
int main(int argc, char* argv[])
{
   unsigned short  tmpa ,tmpb;
   int  re;

re=Regist_Pci8602 () ; //初始化
if(re){ printf("Regist Pci8602 OK !\n") ;                          }
else  { printf("ERRO!\n Press a key to EXIT \n");_getch();return(-1);}

re=DO_Output(0x3aa);          //在数字量输出通道 DO[9..0]输出 0x3aa
re=DI_Input (&tmpa);          //读数字量输入通道 DI[9..0]放入 tmpa

re=Port_WR(PortCFG ,0x82);   // 设置 82C55A 的 PA ,PC 输出;PB 输入
re=Port_WR(PortA,0x55);      // 端口 PA 输出 0x55
re=Port_WR(PortC,0x55);      // 端口 PC 输出 0x55
re=Port_RD(PortB,&tmpa);     // 读端口 PB;

re=PWM_Output(1999,1000);
                      //PWM 输出频率 5KHz ， 占空比 50%
```

```
                        //周期＝（1999＋1）×0.1us =200us  (50KHz)
                        //高电平＝1000×0.1us =100us      （50％）
re=AO_OutputChannel(0,0x8000); //模拟量输出， 0 通道输出 0V；
re=AO_OutputChannel(1,0);      //模拟量输出， 1 通道输出 －10V；
re=AO_OutputChannel(2,0xffff); //模拟量输出， 2 通道输出 ＋10V；

re=Close_Pci8602(); //结束
}
```

7.3 范例程序中断功能;

```
    re=Regist_Pci8602 () ;
    CreateThread(NULL,   0,( unsigned long (__stdcall *)(void *) )StartMe,
                 0,      0,      &pThreadId  );
     printf("\n       Interrupt now be enable !");
     printf("\n       Wait for interrupt..... !");
     printf("\n       Press key to  Exit  demo!\n\n");
     _getch();          //等待中断事件
     re=Close_Pci8602();
     printf("\n       Your interrupt demo  OVER! \n");
     _getch();
     return 0;
}
//StartMe 负责处理中断事件
unsigned long StartMe()
{
 int re ,i=0 ;
 re=InterruptEnable();            //DAS8602  Interrupt Enable
                                  //DI9 triger the interrupt
 while(1)
 {
 re=InterruptWait();              //wait intrrupt
 printf("\n       Interrupt NOW  %x",i++);
 }
    return 0;
}
```

7.3 编程环境设置

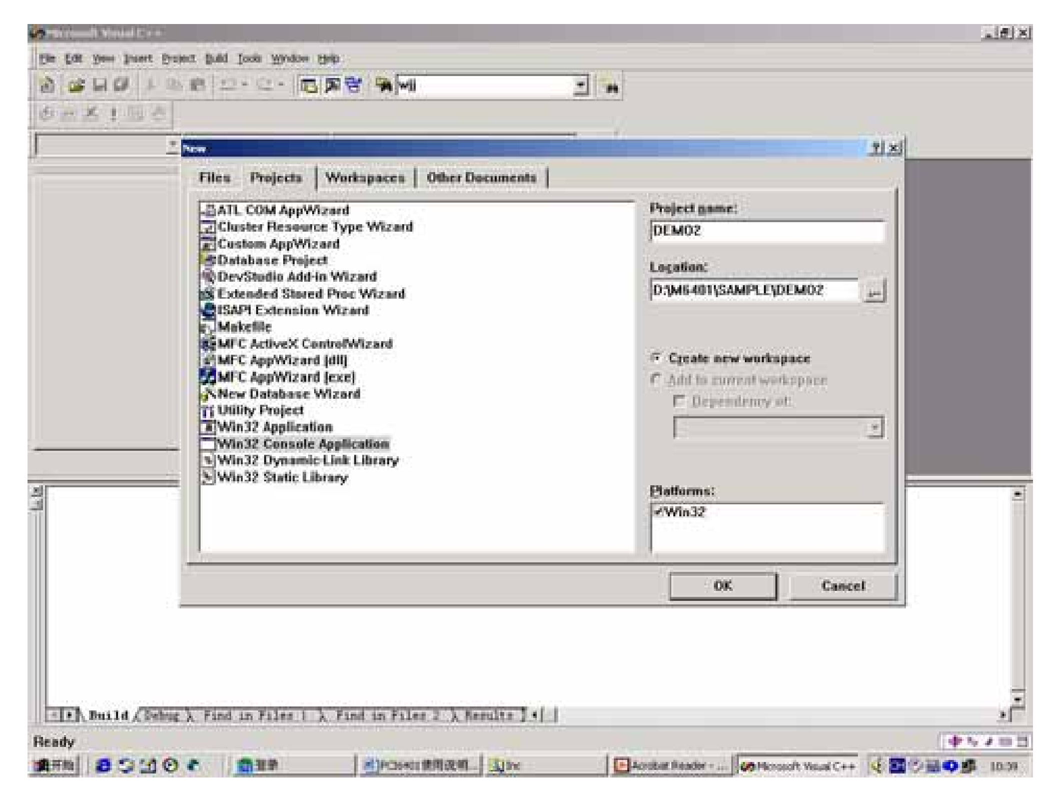
●注意 :在开始编程前，你需要将.\Lib\PlxApi.dll 拷贝到 windows 的 SYSTEM32 目录下，在发布你的最终产品时请在安装文件里这样做。

你的应用程序需要包含以下下两个文件，才能调用如前所述的函数。

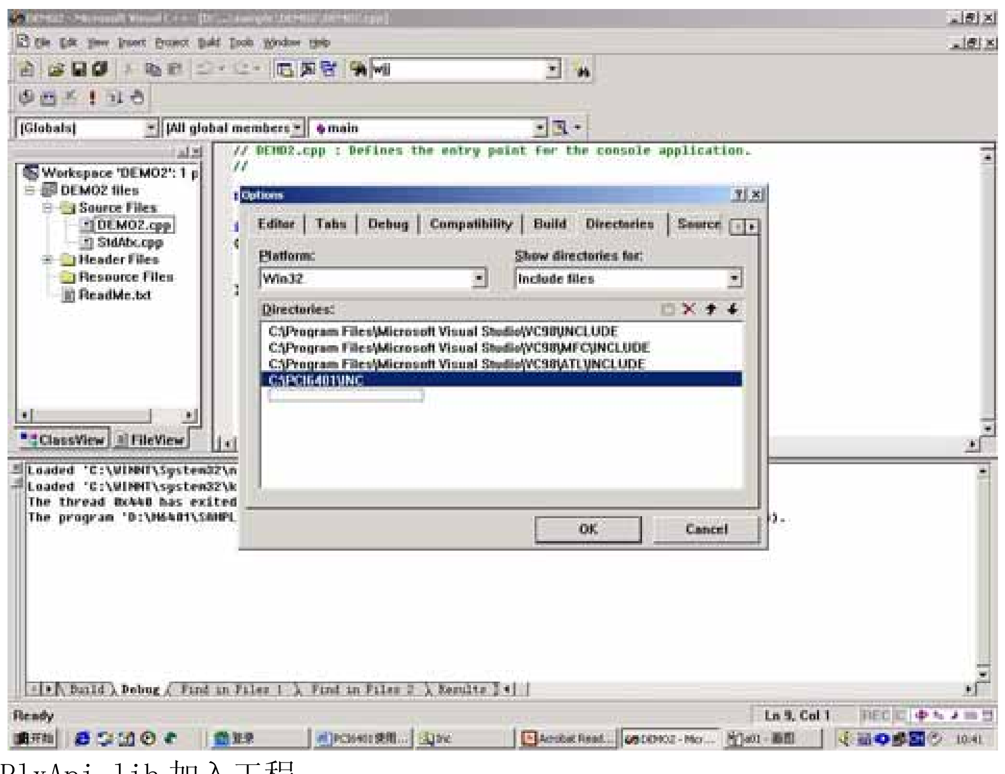
.\Inc\PCI8602.h

.\Lib\PlxApi.lib

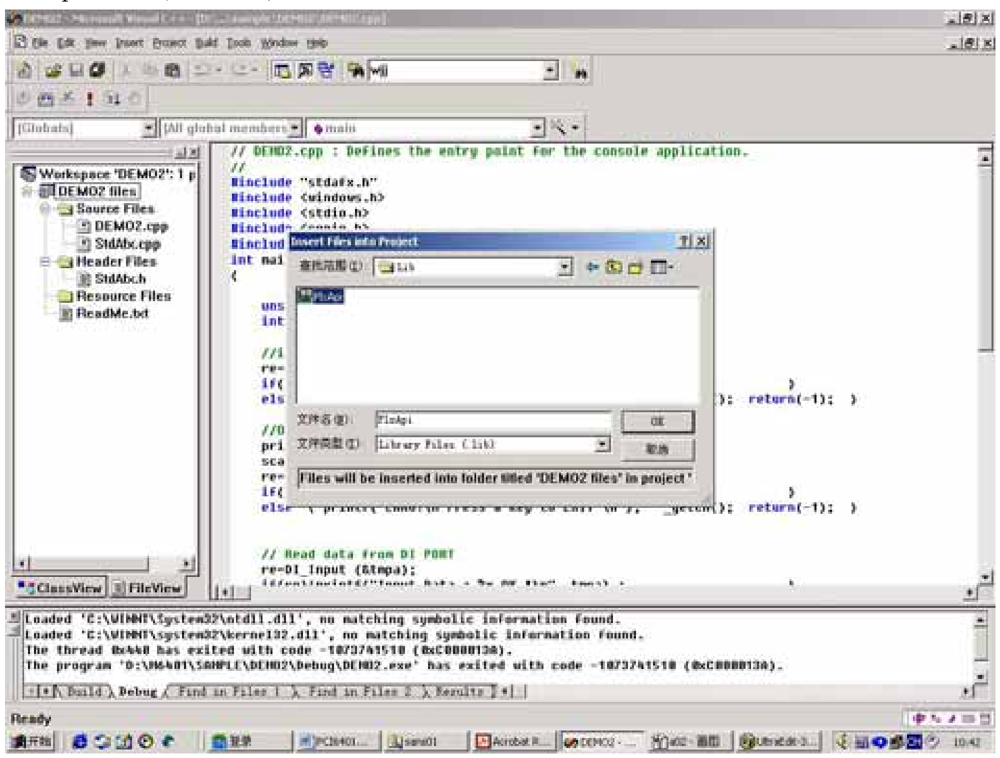
现在我们看如何建立一个新的工程。



将 inc 目录加入



将 PlxApi.lib 加入工程



加入你的代码，现在你可以使用 PCI8602 的功能了.

八、校准

交付到客户的每一块 PCI8601 都经过校准，只有在极其少数的情况下你才有必要重新对板卡进行校准。

8.1 可调电阻

对 PCI8602 各模拟量输出通道的偏移和满度的校准，是通过一组可调电阻进行的，列表如下。

可调电阻编号	功能
VR1	模拟量输出 0 通道偏移校准
VR2	模拟量输出 1 通道偏移校准
VR3	模拟量输出 2 通道偏移校准
VR4	模拟量输出 3 通道偏移校准
VR5	模拟量输出满度校准（＋）
VR6	模拟量输出满度校准（－）

8.2 校准过程

1)模拟通道满度校准;

模拟通道 0 写入 0XFFF,调节 VR5，在输出端得到电压 10V。

模拟通道 0 写入 0X0 ，调节 VR6，在输出端得到电压-10V。

满度校准是所有的输出通道共用，所以只需校准一个通道。

2)模拟量输出 0 通道偏移校准

模拟通道 0 写入 0x8000,调节 VR1，在输出端得到电压 0V。

2)模拟量输出 1 通道偏移校准

模拟通道 1 写入 0x8000,调节 VR2，在输出端得到电压 0V。

2)模拟量输出 2 通道偏移校准

模拟通道 2 写入 0x8000,调节 VR3，在输出端得到电压 0V。

2)模拟量输出 2 通道偏移校准

模拟通道 2 写入 0x8000,调节 VR4，在输出端得到电压 0V。